

Listing of the Claims

In the Claims:

None of the claims are amended in this response, but are listed for the convenience of the Examiner, as follows:

1. (Previously Presented) A programmable graphics pipeline for processing multimedia data comprising:

- (a) an instruction cache for storing at least graphics and media instructions;
- (b) a first register file for storing the multimedia data and intermediate data;
- (c) a first vector functional unit in communication with the instruction cache and the register file, said vector functional unit performing at least graphics and media instructions, to produce at least graphics and media data; and

(d) an enhanced texture cache in communication with the first vector functional unit, the first vector functional unit obtaining from said enhanced texture cache a vector of at least one partition of the multimedia data.

2. (Previously Cancelled)

3. (Previously Presented) The programmable graphics pipeline of Claim 1, wherein the enhanced texture cache comprises:

- (a) a line buffer that provides multiple read ports for accessing the multimedia data; and
- (b) a cache area in communication with the line buffer, said cache area storing the multimedia data.

4. (Original) The programmable graphics pipeline of Claim 3, further comprising an enhanced texture address unit in communication with the enhanced texture cache, said enhanced texture address unit being employed for converting multiple inverse-mapped source coordinates into absolute memory addresses of the multimedia data in an arbitrary-sized block.

5. (Original) The programmable graphics pipeline of Claim 4, wherein the enhanced texture address unit further generates filter coefficients for the multiple inverse-mapped source coordinates for one of a bilinear filtering and a trilinear filtering of the data.

///

///

6. (Original) The programmable graphics pipeline of Claim 4, further comprising an enhanced rasterization unit in communication with the enhanced texture address unit, said enhanced rasterization unit being employed for generating:

- (a) a plurality of destination coordinates for at least one of:
 - (i) a plurality of primitives being rendered; and
 - (ii) a media processing output; and
- (b) a plurality of source coordinates for at least one of:
 - (i) texture data associated with the plurality of destination coordinates for the plurality of primitives; and
 - (ii) media data associated with the plurality of destination coordinates for the media processing.

7. (Original) The programmable graphics pipeline of Claim 6, wherein the plurality of primitives include at least one of:

- (a) dots;
- (b) lines;
- (c) triangles;
- (d) rectangles; and
- (e) polygons.

8. (Original) The programmable graphics pipeline of Claim 6, further comprising an enhanced Z-buffer unit in communication with the enhanced rasterization unit and the enhanced texture address unit, said enhanced Z-buffer unit being employed for loading source coordinates of a primitive being rendered to provide the enhanced texture address unit with variable access to the source coordinates if perspective address generation cannot be used.

9. (Original) The programmable graphics pipeline of Claim 8, wherein the enhanced Z-buffer unit is further employed for determining a depth of a new source pixel in relation to an old source pixel.

10. (Original) The programmable graphics pipeline of Claim 1, further comprising a blending unit that is in communication with the first vector functional unit, said blending unit being employed for:

///

(a) combining a graphics data output from the first vector functional unit with a color value to produce a blended value; and

(b) combining the blended value with a destination pixel value.

11. (Original) The programmable graphics pipeline of Claim 1, further comprising an output buffer in communication with the first vector functional unit, said output buffer being employed for concatenating one of successive image outputs and successive video outputs from the first vector functional unit, to reduce a number of subsequent write transactions to a memory.

12. (Previously Presented) The programmable graphics pipeline of Claim 1, further comprising an output buffer in communication with the first vector functional unit and the enhanced texture cache, said output buffer being employed for:

(a) concatenating successive media processing output from the first vector functional unit to produce concatenated data;

(b) providing the concatenated data as input to the enhanced texture cache; and

(c) converting a compressed format of destination coordinates into a plurality of destination addresses for media processing output data.

13. (Original) The programmable graphics pipeline of Claim 10, further comprising a write buffer in communication with the blending unit, said write buffer sending multiple write transactions to a memory, to reduce page misses.

14. (Original) The programmable graphics pipeline of Claim 11, further comprising a write buffer in communication with the output buffer, said write buffer sending multiple write transactions to a memory, to reduce page misses.

15. (Original) The programmable graphics pipeline of Claim 1, further comprising a configuration register in communication with the first vector functional unit, said configuration register providing the first vector functional unit with partitioned data access parameters and a location of a current instruction thereby instructing the vector functional unit to perform one of a graphics process and a media process.

16. (Original) The programmable graphics pipeline of Claim 1, further comprising:

(a) a second vector functional unit in communication with the instruction cache and performing the graphics and media instructions being performed by the first vector functional unit, to produce parallel graphics and media data; and

(b) a second register file in communication with the second vector functional unit, said second register file storing additional multimedia data and additional intermediate data.

17. (Previously Presented) A method for producing one of graphics pixel data and media output data, comprising the steps of:

(a) obtaining configuration data from a host processor, said configuration data comprising a partitioned data size and a location of an instruction;

(b) performing graphics rendering processing on graphics texture data with a programmable graphics rendering pipeline to produce graphics pixel data when the partitioned data size and the instruction correspond to graphics processing; and

(c) performing media processing on media source data with the programmable graphics rendering pipeline to produce media output data when the partitioned data size and the instruction correspond to media processing.

18. (Previously Presented) The method of Claim 17, wherein the step of performing graphics rendering processing comprises the steps of:

(a) producing a destination coordinate;

(b) producing a mapped texture coordinate corresponding to the destination coordinate;

(c) producing a plurality of surrounding memory addresses in which surrounding texture data are stored, said surrounding texture data corresponding to a plurality of surrounding texture coordinates surrounding the mapped texture coordinate;

(d) retrieving the surrounding texture data from the plurality of surrounding memory addresses, and storing the surrounding texture data retrieved in a cache; and

(e) performing interpolation on the surrounding texture data in the cache to determine a texture value at the mapped texture coordinate as the graphics pixel data for the destination coordinate.

19. (Original) The method of Claim 17, wherein the step of performing media processing corresponds to image processing comprises the steps of:

(a) producing a destination coordinate;

(b) producing at least one mapped image coordinate corresponding to the destination coordinate;

1 (c) producing at least one memory address where image data are stored
2 corresponding to the at least one mapped image coordinate;

3 (d) retrieving the image data from the at least one memory address, and storing the
4 image data retrieved in a cache; and

5 (e) performing an image manipulation function on the image data in the cache to
6 produce image pixel data at the destination coordinate as the media output data.

7 20. (Original) The method of Claim 17, wherein the step of performing media processing
8 corresponds to video processing comprises the steps of:

9 (a) producing a destination coordinate;

10 (b) producing at least one mapped video coordinate corresponding to the
11 destination coordinate;

12 (c) producing at least one memory address where video data are stored
13 corresponding to the at least one mapped image coordinate;

14 (d) retrieving the video data from the at least one memory address, and storing the
15 video data retrieved in a cache; and

16 (e) performing a video manipulation function on the video data to produce video
17 pixel data at the destination coordinate as the media output data.

18 21. (Previously Presented) The method of Claim 17, wherein the programmable graphics
19 rendering pipeline is employed for executing at least one of a partitioned inner product instruction, a
20 partitioned arithmetic instruction, a partitioned logic instruction, a data movement instruction, and a
21 loop-control instruction, to produce at least one of the graphics output pixel data and the media output
22 data.

23 22. (Previously Presented) The method of Claim 17, further comprising the step of storing
24 the one of the graphics pixel data and the media output data in memory.

25 23. (Previously Presented) A programmable graphics pipeline for multimedia applications
26 that performs graphics and media functions, comprising:

27 (a) a vector streaming engine that accesses variable length data from a memory
28 and writes pixel data to the memory, wherein the variable length data comprise one of graphic texture
29 source data and media source data, and wherein the pixel data are one of graphics pixel data and
30 media pixel data; and

(b) a vector processing engine in communication with the vector streaming engine, said vector processing engine generating the pixel data from the variable length data.

24. (Original) The programmable graphics pipeline of Claim 23, wherein the vector streaming engine comprises:

(a) a vector input unit in communication with the vector processing engine; and

(b) a vector output unit in communication with the vector processing engine.

25. (Previously Presented) The programmable graphics pipeline of Claim 24, wherein the vector input unit comprises:

(a) an enhanced rasterization unit that generates at least one of:

(i) a destination coordinate corresponding to one of a graphics pixel location and a media pixel location; and

(ii) a source coordinate for one of a graphics texture source, and a media source; and

(b) an enhanced texture address unit in communication with the enhanced rasterization unit, said enhanced texture address unit generating a memory address corresponding to the source coordinate, said memory address defining a storage location for one of the graphics texture source data and the media source data; and

(c) a texture cache in communication with the enhanced texture address unit, said texture cache storing said one of the graphics texture source data and the media source data for use by the vector processing engine.

26. (Original) The programmable graphics pipeline of Claim 24, wherein the vector output unit comprises:

(a) a blending unit in communication with the vector processing engine, said blending unit performing blending operations on graphics pixel data;

(b) an output buffer in communication with the vector processing engine, said output buffer being employed for concatenating a plurality of media pixel data; and

(c) a write buffer in communication with both the blending unit and the output buffer, said write buffer writing one of the graphics pixel data and the media pixel data to memory in burst groups.

///

27. (Previously Presented) The programmable graphics pipeline of Claim 23, wherein the vector processing engine comprises:

(a) a first vector functional unit in communication with the vector streaming engine, said first vector functional unit being usable for graphics rendering processing functions and media processing functions;

(b) an instruction cache in communication with the vector functional unit, said instruction cache storing at least one machine instruction to be executed by the vector functional unit; and

(c) a first register file in communication with the first vector functional unit, said first register file storing intermediate data for use by the first vector functional unit.

28. (Original) The programmable graphics pipeline of Claim 27, wherein the vector processing engine comprises:

(a) a second vector functional unit in communication with the vector streaming engine, and in communication with the instruction cache, said second vector functional unit being usable for performing graphics processing functions and media processing functions in parallel with the first vector functional unit; and

(b) a second register file in communication with the second vector functional unit, said second register file storing intermediate data for use by the second vector functional unit.

29. (Original) The programmable graphics pipeline of Claim 23, further comprising a configuration register in communication with both the vector streaming engine and the vector processing engine, said configuration register providing a partitioned data size and a location of an instruction indicating operation of the programmable pipeline for one of graphics processing and media processing.

30. (Previously Presented) A method for performing one of a graphics function and a media function on variable length data with a programmable graphics pipeline, comprising the steps of:

(a) using a vector streaming engine:

(i) generating a plurality of output pixel coordinates;

(ii) generating a plurality of memory addresses corresponding to the output pixel coordinates at which are stored one of graphics texture data and media source data, wherein the graphics texture data and the media source data are not of equal length; and

1 (iii) caching said one of the graphics texture data and the media source data
2 in a cache;
3 (b) passing data retrieved from the cache to a vector processing engine; and
4 (c) performing one of the graphics function and the media function on the data
5 retrieved from the cache using the vector processing engine.
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30